Politechnika Gdańska

Wydział Elektrotechniki i Automatyki

Gdańsk 2015

ANALIZA WPŁYWU TRANSMISJI DANYCH NA JAKOŚĆ DZIAŁANIA ROZPROSZONYCH SYSTEMÓW STEROWANIA

Przemysław SPYCHALSKI[1], Ryszard ARENDT[2]

Politechnika Gdańska, Wydział Elektrotechniki i Automatyki
1. tel.: 58 347 2139, e-mail: przspych@student.pg.gda.pl
2. tel.: 58 347 2157, e-mail: ryszard.arendt@pg.gda.pl

Streszczenie: W artykule przedstawiono zagadnienia związane z transmisją danych w rozproszonych systemach sterowania. Opisano cechy charakterystyczne przemysłowych sieci i protokołów informatycznych. Utworzono model rozproszonego systemu sterowania poziomem cieczy w zbiorniku, w którym komunikacja oparta została na protokole Modbus RTU. Na podstawie zmian parametrów transmisji danych wykazano, że poprawna komunikacja pomiędzy elementami systemu ma kluczowe znaczenie w zapewnieniu odpowiedniej jakości sterowania.

Słowa kluczowe: transmisja danych, komunikacja, rozproszony system sterowania, Modbus RTU.

1. WPROWADZENIE

Rozwój technologii sieciowych spowodował decentralizację układów sterowania dzięki możliwości przesyłania danych na odległość. Sterowanie lokalne zastąpione zostało sterowaniem zdalnym. W zakładach produkcyjnych, elektrowniach, rafineriach pojawiły się sterownie oraz stacje operatorskie umożliwiające monitoring procesu produkcyjnego i jego kontrole z dystansu. Podejście to niosło za sobą wiele korzyści, jednak przesyłanie dużej ilości danych procesowych w skończonym czasie stanowiło trudne wyzwanie [1].

Informacje przesyłane w systemach sterowania są niezwykle istotne, dlatego utrata pakietów danych lub przerwanie transmisji może mieć bardzo negatywne skutki. Standardowe protokoły wymiany informacji, takie jak TCP/IP lub UDP nie były przystosowane do deterministycznej komunikacji pomiędzy elementami systemów sterowania i kontroli poprawności transmisji. Powstały zatem dedykowane rozwiązania różniące się od powszechnie znanych standardów transmisji danych [2, 3]. W artykule przedstawiono dwa z nich: Profibus i Modbus RTU oraz zbadano wpływ zmiany parametrów transmisji na jakość sterowania w przykładowym systemie rozproszonym.

2. TRANSMISJA DANYCH W SYSTEMACH PRZEMYSŁOWYCH

2.1. Informacje ogólne

Pod koniec XX wieku zaimplementowane zostały protokoły komunikacyjne określające sposób transmisji danych w rozproszonych systemach sterowania.

W przemysłowych sieciach informatycznych bardzo ważny jest determinizm czasowy transmisji danych oraz ustandaryzowany sposób dostarczania kolejnych pakietów informacji. Przewidywalność procesu komunikacji w rozproszonych systemach sterowania warunkuje prawidłowy nadzór i kontrole nad procesem technologicznym. Często spotykana jest również redundancja połączeń w celu utrzymania transmisji, kiedy jedno z mediów komunikacyjnych zawiedzie. Przykładem może być przesyłanie informacji procesowych za pośrednictwem łącza światłowodowego oraz nadmiarowo drogą bezprzewodową (GPRS, Radiomodemy).

Transmisja danych w systemach przemysłowych odbywa się w czasie rzeczywistym. W celu weryfikacji jej poprawności stosowane są sumy kontrolne obliczane dla każdej przesyłanej ramki danych pozwalające sprawdzić, czy informacje przesyłane w ramce nie zostały uszkodzone. Wszystkie wcześniej opisane cechy znalazły swoje odzwierciedlenie w przemysłowej sieci informatycznej Profibus oraz protokole komunikacyjnym Modbus RTU.

2.2. Profibus

Przykładem otwartej, modułowej sieci komunikacyjnej stosowanej w automatyce przemysłowej jest Profibus. Bazuje na modelu sieciowym ISO/OSI, wykorzystując warstwę fizyczną, łącza danych oraz aplikacji tego modelu. Profibus współpracuje najczęściej z przewodowym medium transmisyjnym RS-485. Omawiana sieć przemysłowa oparta jest na architekturze Master/Slave. Przykładowo urządzeniem nadrzędnym w procesie komunikacji (Master) może być sterownik PLC, natomiast urządzeniem podrzędnym (Slave) moduł wejść/wyjść. Maksymalna prędkość transmisji w tej sieci przemysłowej to 12 Mb/s.

Występują trzy podstawowe wersje sieci Profibus w zależności od profilu zastosowań: Profibus DP (ang. Decentralised Peripherals), Profibus PA (ang. Process Automation) oraz PROFIsafe. Pierwsza wersja sieci Profibus stosowana jest w systemach sterowania do transmisji danych pomiędzy sensorami i aktuatorami, a sterownikiem. Druga wersja stosowana jest w automatyce procesowej do zdalnego monitorowania urządzeń wchodzących w skład systemu sterowania. Ostatni wariant sieci Profibus używany jest do przesyłania danych w automatyce zabezpieczeniowej [1, 3, 4].

3. BUDOWA ROZPROSZONEGO SYSTEMU STEROWANIA

W celu zbadania wpływu zmian parametrów transmisji na jakość sterowania utworzono rozproszony system sterowania poziomem cieczy w zbiorniku prostopadłościennym, w którym poziom regulowany był za pomocą dwóch pomp (doprowadzającej oraz odprowadzającej ciecz ze zbiornika). Zaproponowany obiekt sterowania był obiektem inercyjnym, opisywało go następujące równanie transmitancyjne w dziedzinie dyskretnej [5]:

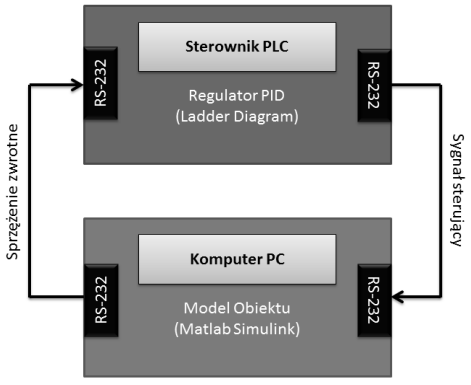
$$G(z) = \frac{0{,}6321}{z - 0{,}3679}$$

gdzie: $G(z)$ – transmitancja operatorowa,
z – zmienna zespolona

Model obiektu sterowania zaimplementowany został w programie Matlab z wykorzystaniem biblioteki SDRT (ang. Simulink Desktop Real-Time). Narzędzie to pozwala na wykonywanie symulacji w czasie rzeczywistym oraz umożliwia komunikacje w pętli zamkniętej pomiędzy symulowanym obiektem, a fizycznym urządzeniem sterującym, na przykład PLC/PAC. Taki rodzaj symulacji określa się mianem HIL (ang. Hardware-In-The-Loop) i wykorzystuje do szybkiego prototypowania rozproszonych systemów sterowania.

Symulacja procesu regulacji poziomu cieczy odbywała się po stronie środowiska Matlab Simulink na komputerze PC, dzięki czemu możliwy był podgląd aktualnego stanu wartości regulowanej. Algorytm sterowania PID (ang. Proportional-Integral-Derivative) zaimplementowany został natomiast po stronie sterownika PLC/PAC w języku drabinkowym. Nastawy regulatora PID dobrane zostały automatycznie z wykorzystaniem wbudowanych funkcji sterownika (PID Autotuning).

Ze sterownika PLC wysyłany był sygnał sterujący, natomiast z komputera PC sprzężenie zwrotne o aktualnym stanie poziomu cieczy w zbiorniku. Regulator PID w sterowniku na podstawie uchybu regulacji wyliczał kolejną wartość sygnału sterującego i proces się powtarzał. Transmisja danych pomiędzy komputerem PC, a sterownikiem PLC odbywała sie z wykorzystaniem łącza szeregowego RS-232. Schemat transmisji danych pomiędzy sterownikiem PLC, a komputerem PC w technologii Hardware-In-The-Loop przedstawiono na rysunku 1 [5].



Rys. 1. Schemat transmisji danych pomiędzy sterownikiem PLC, a komputerem PC w technologii Hardware-In-The-Loop [5]

4. IMPLEMENTACJA PROTOKOŁU KOMUNIKACYJNEGO MODBUS RTU

Transmisja danych w utworzonym systemie sterowania oparta została na protokole Modbus RTU (ang. Remote Terminal Unit). Jest to otwarty protokół przemysłowy, w którym dane przesyłane są w postaci binarnej. Modbus opracowany został przez firmę Modicon i stał się jednym z podstawowych protokołów używanych do komunikacji pomiędzy urządzeniami w systemach sterowania.

Działa on w trybie Master/Slave, przy czym standardowo w sieci występuje jedno urządzenie nadrzędne (Master) oraz do 247 urządzeń podrzędnych (Slave). Każde urządzenie Slave posiada swój własny, unikalny identyfikator (SlaveID). Master jest urządzeniem inicjalizującym proces komunikacji. Wysyła jako pierwszy żądanie do urządzeń typu Slave. Początkowy bajt ramki danych zawiera unikatowy SlaveID, dzięki czemu komunikacja nawiązana zostaje z właściwym urządzeniem podrzędnym (pozostałe urządzenia ignorują ramki danych, które nie są do nich adresowane).

Dzięki temu mechanizmowi transmisja danych w sieci przebiega w sposób uporządkowany. Kolejny bajt ramki danych zawiera kod funkcji, czyli informacje dla urządzenia Slave jaką akcję powinien wykonać. Może być to na przykład zapis lub odczyt rejestrów, wejść bitowych oraz odczyt stanu urządzenia Slave. Kolejne bity ramki zawierają dane dodatkowe związane z operacją wskazaną przez kod funkcji, przykładowo adres rejestru, do którego ma nastąpić zapis oraz wartość jaka ma zostać zapisana.

W podstawowej wersji protokołu Modbus RTU dane zapisać można na 8 bitach, co daje możliwość przesłania wartości w zakresie 0-255, jednak najczęściej stosowana jest modyfikacja pozwalająca na zwiększenie zakresu o N x 8 bitów. Każda ramka danych opatrzona jest sumą kontrolną CRC (ang. Cyclic Redundancy Check). Suma kontrolna obliczana jest dla każdej wysyłanej i odbieranej ramki, dzięki czemu następuje weryfikacja poprawności przesyłania danych.

Oprócz tego możliwa jest diagnostyka determinizmu czasowego transmisji, ponieważ pomiędzy przesyłanymi bajtami danych w ramce wymagana jest przerwa o długości 1.5 CHAR (gdzie CHAR oznacza czas transferu pojedynczego znaku, czyli 8 bitów). Znacznikiem wysłania całej ramki jest pauza trwająca co najmniej 3.5 CHAR. Przekroczenie tych wartości skutkuje zwróceniem kodu błędu transmisji. Postać ramki danych protokołu Modbus RTU przedstawiona została na rysunku 2 [4, 5, 6].

Start	SlaveID	Function	Data	CRC	End
>= 3.5 CHAR	8 bits	8 bits	N x 8 bits	16 bits	>= 3.5 CHAR

Rys. 2. Postać ramki danych protokołu Modbus RTU [6]

W środowisku Matlab Simulink zaimplementowana została programowa realizacja prokokołu Modbus RTU. W dalszej części przedstawiono fragment kodu odpowiadający za enkapsulację ramki danych, dzięki której możliwy był zapis wartości regulowanej do rejestru %R0001 sterownika PLC. Numeracja rejestrów rozpoczyna się od zera, co uwzględniono przy podawaniu adresu rejestru początkowego (nastąpiło przesunięcie o N-1). Struktura ramki danych zapisu znajduje się na rysunku 3 [5].

```
//Dyskretne równanie stanu obiektu sterowania
z = 0.6321 + 0.3679 * z;
//Dekompozycja wartości regulowanej na dwa bajty
       DATA_1 = floor(z/256);
       DATA_2 = mod(z,256);
//Algorytm obliczania sumy kontrolnej CRC
writeFrame = [01 06 00 00 DATA_1 DATA_2]
N = length(writeFrame);
crc = hex2dec('ffff');
polynomial = hex2dec('a001');

for i = 1:N
    crc = bitxor(crc,message(i));
    for j = 1:8
        if bitand(crc,1)
            crc = bitshift(crc,-1);
            crc = bitxor(crc,polynomial);
        else
            crc = bitshift(crc,-1);
        end
    end
end

CRC_1 = bitand(crc,hex2dec('ff'));
CRC_2 = bitshift(bitand(crc,hex2dec('ff00')),-8);
//Enkapsulacja ramki danych zapisu do rejestru
writeFrame = [01 06 00 00 DATA_1 DATA_2 CRC_1 CRC_2]
```

Start	SlaveID	Function	Data	CRC	End
>= 3.5 CHAR	01	06	DATA_1 DATA_2	CRC_1 CRC_2	>= 3.5 CHAR

Rys. 3. Ramka danych zapisu wartości regulowanej do rejestru sterownika

Wartość regulowana i suma kontrolna rozbite zostały na dwa bajty, co odpowiada kolejnym polom w ramce: DATA_1, DATA_2, CRC_1, CRC_2. Algorytm obliczania sumy kontrolnej CRC pochodził ze strony internetowej MathWorks [7]. Na podstawie otrzymanej ramki danych regulator PID zaimplementowany w sterowniku obliczał wartość sygnału sterującego, którą następnie zapisywał do sąsiedniego rejestru sterownika %R0002. Aby wartość sygnału sterującego dotarła do modelu obiektu, komputer PC (Master) przesyłał do sterownika ramkę danych z żądaniem odczytu rejestru %R0002 (rys. 4) [5]:

```
readFrame = [01  03  00  01  00  01  213 202]
```

Start	SlaveID	Function	Data	CRC	End
>= 3.5 CHAR	01	03	00 01 00 01	213 202	>= 3.5 CHAR

Rys. 4. Ramka danych odczytu wartości sygnału sterującego z rejestru sterownika

Przygotowane ramki danych przesyłane były za pośrednictwem portu szeregowego RS-232. Matlab Simulink posiada wbudowane komendy do obsługi portu transmisji szeregowej, między innymi: fopen, fwrite, fread, fclose. W poleceniu fread jako parametr należało podać liczbe bitów jaka powinna zostać odczytana. Możliwa była również konfiguracja parametrów transmisji podczas definiowania obiektu portu, takich jak: numer fizyczny portu, kontrola determinizmu czasowego, ilość przesyłanych bitów danych, bit stopu, szybkość transmisji, kontrola przepływu. Aby umożliwić transmisję danych pomiędzy elementami systemu sterowania odpowiednio skonfigurowano port szeregowy oraz użyto wcześniej wymienionych funkcji do wysyłania ramek danych przez RS-232.

Kod programu realizujący komunikację pomiędzy komputerem PC i sterownikiem PLC za pośrednictwem portu szeregowego przedstawiono poniżej [5]:

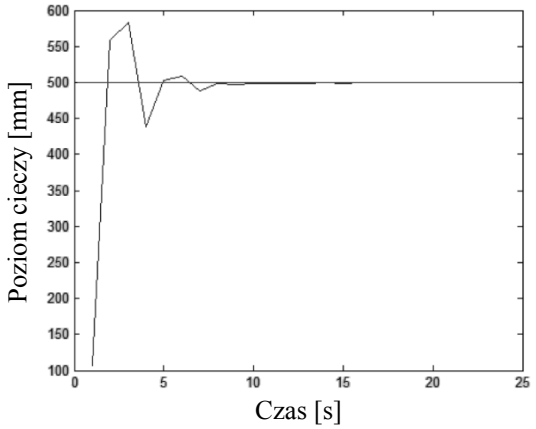
```
port = serial
('COM3','BaudRate',9600,'DataBits',8,'StopBits',1,
'Timeout',1,'Parity','none')
fopen (port)
fwrite(port,whiteFrame)
controlledValue = fread(port,8)
fwrite(port,readFrame)
controlSignalPartial = fread(port,7)
controlSignal = controlSignalPartial(4,1) * 256 +
controlSignalPartial(5,1)
fclose (port)
```

Całość kodu programu wykonywana była w pętli, aby zapewnić płynną regulację poziomu cieczy w zaimplementowanym zbiorniku. Po odebraniu ramki z aktualną wielkością przechowywaną w rejestrze sterownika należało dokonać przekształcenia, aby odczytać faktyczną wartość sygnału sterującego wygenerowaną przez regulator PID. Początkowa prędkość transmisji danych wynosiła 9600 b/s. Zdecydowano na podstawie zmiany tego atrybutu oszacować wpływ zmiany parametrów transmisji na jakość sterowania. Kolejne pomiary wykonane zostały odpowiednio dla 1200 b/s oraz 600 b/s. Na komputerze PC z zaimplementowanym modelem zbiornika możliwy był podgląd trajektorii sterowania w programie Matlab Simulink i oszacowanie jak dana prędkość transmisji wpływa na takie parametry regulacji jak czas ustalania, przeregulowanie, oscylacje wokół wartości zadanej, uchyb w stanie ustalonym. Wyniki badań przedstawiono w kolejnym paragrafie.

5. WPŁYW PARAMETRÓW TRANSMISJI NA JAKOŚĆ STEROWANIA

Standardowa wartość natężenia strumienia danych stosowana w przypadku protokołu Modbus RTU to 9600 b/s. Uzyskano dla niej przeregulowanie na poziomie 15%, czas ustalania około 7 sekund, nie występowały oscylacje wokół wartości zadanej (rys. 5).

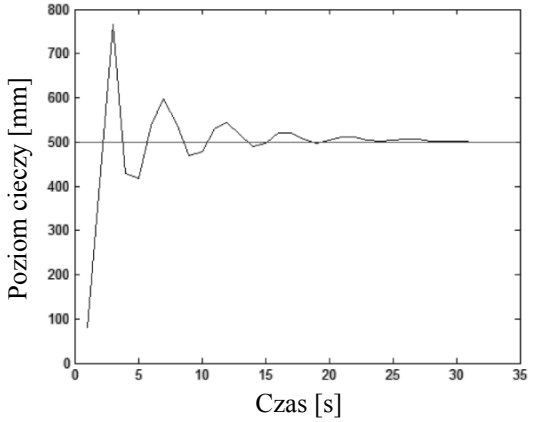
```
port = serial
('COM3','BaudRate',9600,'DataBits',8,'StopBits',1,
'Timeout',1,'Parity','none')
```



Rys. 5. Trajektoria sterowania dla prędkości transmisji 9600 b/s

Dla prędkości transmisji wynoszącej 1200 b/s stwierdzono wydłużenie czasu regulacji, wzrost przeregulowania oraz pojawienie się oscylacji wokół wartości zadanej (rys. 6).

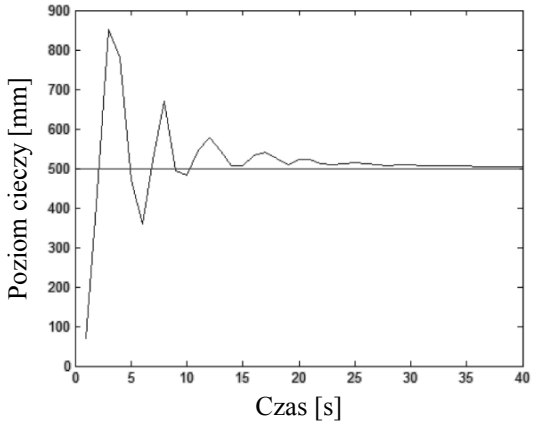
```
port = serial
('COM3','BaudRate',1200,'DataBits',8,'StopBits',1,
'Timeout',1,'Parity','none')
```



Rys. 6. Trajektoria sterowania dla prędkości transmisji 1200 b/s

Po zmianie prędkości transmisji na 600 b/s nastąpiło dalsze pogorszenie jakości sterowania objawiające się zwiększeniem przeregulowania do blisko 60%, dłuższym czasem ustalania i uchybem w stanie ustalonym (rys. 7).

```
port = serial
('COM3','BaudRate',600,'DataBits',8,'StopBits',1,
'Timeout',1,'Parity','none')
```



Rys. 7. Trajektoria sterowania dla prędkości transmisji 600 b/s

Pogorszenie się jakości sterowania podczas zmniejszania szybkości transmisji spowodowane było wydłużeniem czasu oczekiwania na zmiane wartości sygnału sterującego, który obliczany był w sterowniku PLC. Przykładowo opóźnienie jakie generował port RS-232 podczas przesyłania ramki danych przy prędkości transmisji 9600 b/s wynosiło 5.1 ms, jednak dla 1200 b/s i 600 b/s było to odpowiednio 40.8 oraz 81.7 milisekund.

6. PODSUMOWANIE

Dobór prawidłowych parametrów transmisji danych w rozproszonych systemach sterowania jest kluczowy dla zachowania odpowiedniej jakości kontroli nad procesem. Wykazano, że ustawienie zbyt niskiego współczynnika natężenia strumienia danych powoduje znaczne pogorszenie przebiegu trajektorii sterowania.

Podczas implementacji protokołów komunikacyjnych wykorzystywanych w rozproszonych systemach sterowania należy wdrożyć mechanizmy weryfikujące poprawność transmisji, takie jak sumy kontrolne CRC. Transmisja danych w rozproszonych systemach sterowania powinna przebiegać w sposób uporządkowany z zachowaniem odpowiedniego determinizmu czasowego.

Często stosowanym modelem komunikacji w automatyce jest architektura Master/Slave, która zakłada, że jedno z urządzeń biorących udział w procesie wymiany informacji jest nadrzędne i zarządza wysyłaniem komunikatów do urządzeń podrzędnych.

7. BIBLIOGRAFIA

1. Naskręt P., Bojanowski W., Pawlak K.: Nowoczesne systemy transmisji danych w automatyce przemysłowej, VIII Konferencja Odlewnicza Technical, Nowa Sól 2005
2. Arendt R., Michalski R.: Fault diagnosing system of wheeled tractors, LAP LAMBERT Academic Publishong, Saarbrücken, Deutchland 2015, p. 115
3. Sen S. K.: Fieldbus and Networking in Process Automation, CRC Press 2014, p. 461
4. Lisowski G.: Przemysłowe interfejsy komunikacyjne, Politechnika Łódzka, Łódź 2006
5. Spychalski P.: Komunikacja w układach sterowania z wykorzystaniem radiomodemów ethernetowych, Praca inżynierska, Politechnika Gdańska, Gdańsk 2013
6. MODBUS over Serial Line Specification and Implementation Guide v1.02
7. http://www.mathworks.com/matlabcentral/fileexchange/36022-rs485-modbus-communication-with-jld416pva-power-meter/content/append_crc.m

IMPACT ANALYSIS OF DATA TRANSMISSION ON EFFICIENCY OF DISTRIBUTED CONTROL SYSTEMS

This paper describes common issues related to data transmission in distributed control systems. Main features of industrial networks and communication protocols were presented. Model of distributed control system liquid level in the tank was developed, where data transmission was based on Modbus RTU protocol. Results of changing data transmission parameters proved that correct communication between system components is essential in ensuring proper control quality.

Keywords: data transmission, communication, distributed control system, Modbus RTU.